LA-UR-17-28311
Approved for public release; distribution is unlimited.

Title: Software Infrastructure for V&V

Author(s): Israel, Daniel M.

Intended for: Physics V&V L2 Milestone Review

Issued: 2017-09-14

# Software Infrastructure for V&V

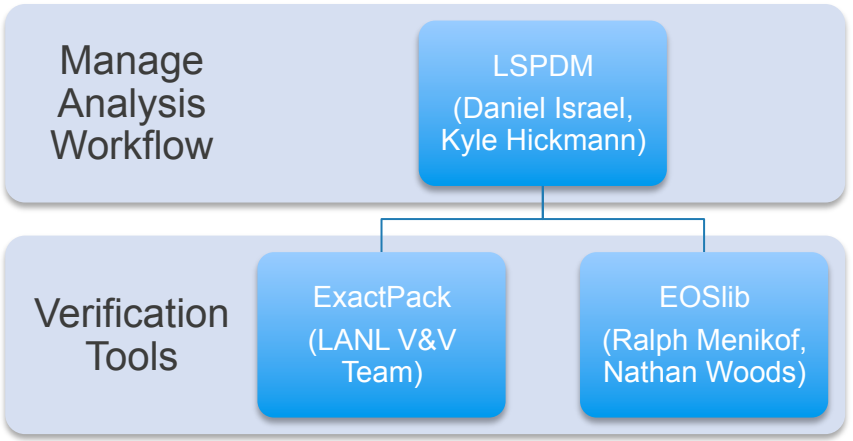
## Physics V&V L2 Milestone Review


**Speaker: Daniel Israel**

NNSA
National Nuclear Security Administration
Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA

# Overview

- **Historically, V&V practitioners have developed their own tools.**
  - Significant redundancy in effort
  - Difficult to maintain
  - Individual tools are not corporate knowledge
- **In this milestone, the V&V team undertook a move toward standardized, shared tools.**
- **This necessitated identifying requirements.**
- **Today we will talk about three of those tools.**



# Verification infrastructure requirements

- *Scenario: We have a verification document, but it is not clear how some details of the case were set up.*
- **Requirement**: All input files needed to recreate the case must be stored.
- *Scenario: We have input and output files, but it is not clear if they are all from the same run.*
- **Requirement**: System must store pedigree of all output files.
- *Scenario: Accreditation requires duplicating studies for each new code release.*
- **Requirement**: Suites must be easy to repeat.
- *Scenario: Something to have changed in some results, and we are not sure why.*
- **Requirement**: Results must be captured for a clear historical record.
- *Scenario: A colleague who is not an SME wants to work with the data.*
- **Requirement**: Studies must be easy to share and reproduce.

# Verification results must stay fresh

- **Verification cannot be a "vet it and forget it" activity.**
  - Why?
- **Each new code release may include:**
  - New code options, parameter settings, and defaults, all of which must be tested,
  - Improvements to algorithms which can change convergence properties,
  - New bugs.
- **In a complex multi-physics code, changes in one part of the code may have unexpected results elsewhere.**

***Verification tests must be regularly repeated.***

# EOSlib

- **Requirement: An open-source software package for analytic thermodynamic models**
  - A single target for verification efforts involving thermodynamic models.
  - Easy access to different thermodynamic models for ExactPack solvers.
  - Availability of a reference implementation of thermodynamic models for the purposes of code-to-code
- **Solution: EOSLib**
  - Developed by Ralph Menikoff
  - We have repackaged it for easy installation and use, and are working toward a Python interface (Nathan Woods & William Magrogan)
- **Available on Stash and Github**
  - xcp-stash.lanl.gov/projects/PHYSVER/repos/eoslib
  - www.github.com/lanl/EOSlib

# EOSlib Features

- **Extensible, stand-alone code for analytic EOS models.**
- **Provides a single target for verification and validation work.**
- **Provides tools for managing databases of EOS parameters.**
- **Facilitates EOS computations:**
  - Hugoniot curves
  - Detonation profiles
  - Simple plastic deformation

***Our eventual plan is to use EOSlib for all EOS calculations in ExactPack.***

# The EOSlib Database

- Human-readable data files
- Tools for searching, querying database files
- Built-in calculator and unit conversions
- Easy links to out-of-source data files and libraries

```
EOS:IdealGas=7/5; units=hydro::CGS
{
   gamma = 7/5
   V_ref = 1/1.275                          # rho = 1.275 g/cm^3
   e_ref = 1.013e5 * V_ref / (gamma – 1)    # P = 1.013e5
   Cv    = e_ref/300                        # Tref = 300 K
}
```

# EOSLib included applications

| | |
|---|---|
| Wave | • Easy interface to basic hydrodynamic waves.<br>• Compute Hugoniot loci, CJ detonations, etc. |
| ZND | • Compute ZND profile using arbitrary rate law. |
| ImpedanceMatch | • Riemann solver for general EOS. |
| Polar | • Compute shock polar |

# ExactPack, making verification easy

**The old way of doing it:**

- **Find an exact solution code for the problem of interest**
  - Look online
  - Ask a collegue
  - Write one yourself
- **Run the physics solver code on various grids**
- **Create a reader to import code data and another for exact solution data**
- **Write a script to perform the convergence analysis**

**Using ExactPack:**

- **Run the physics solver code on various grids**
- **Use ExactPack to read the data, compute the exact solution, compare the data, do a convergence analysis, and plot the results.**

# ExactPack

- **ExactPack (LA-CC-14-047) is a Python package providing a common interface to a wide variety of exact solutions to physics problems.**
- **Use as stand-alone code or write Python scripts that call exact solvers.**
- **Easily extensible**
  - Add your own solvers
  - Solvers can be written in Python, Fortran, or C
- **Rigourous unit testing for reliability**
- **Extensive documentation available in HTML or PDF (LA-UR-16-23260r1)**
- **ExactPack also contains a library of analysis tools for code verification (convergence analysis).**
- **Collaborative model, i.e. contribute back your own solver via git**
  - xcp-stash.lanl.gov/projects/PHYSVER/repos/exactpack
  - www.github.com/lanl/ExactPack

# Available Solvers

| Pure Hydrodynamics | Conduction | Shock Hydrodynamics with Radiation | High Explosives | Materials |
|---|---|---|---|---|
| •Riemann Shock Tubes (6 problem variants)<br>•Sod, Einfeldt, Stationary-Contact, Slow-Shock, Shock-Contact-Shock, LeBlanc<br>•Riemann with the JWL EOS (2 variants)<br>•Noh<br>•Noh's second problem (uniform collapse)<br>•Sedov<br>•Guderley | •Heat conduction problems<br>•Coggeshall problems (20 of the 22 problems)<br>•Reinicke Meyer-ter-Vehn<br>•Su-Olson | •Lowrie-Rauenzahn equilibrium diffusion<br>•Lowrie-Edwards non-equilibrium diffusion<br>•Ferguson-Lowrie-Morel grey Sn transport | •Mader<br>•Escape of High Explosives Problem<br>•Steady Domain Reaction Zone Problem<br>•Kenamond/ Programmed Burn<br>•Detonation Shock Dymamics<br>•Steady Zel'dovich-von Neumann-Döring (ZND) wave tests for arbitrary reaction rate law and equation of state (coming soon) | •Blake<br>•Hunter |

# A sample convergence analysis in ExactPack

```
import glob
import matplotlib.pylab as plt

from exactpack.solvers.sedov import Sedov
from exactpack.analysis import Study, \
     RegressionConvergenceRate, CellNorm
from lanl_readers.rage import RageSpherical

study = Study(sorted(glob.glob('res*/sedov_1d-dmp*')),
              reference=Sedov(),
              study_parameters=[0.00625, 0.003125, 0.0015625, 0.00078125, 0.000390625],
              time=1.0,
              reader=RageSpherical(),
              abscissa = 'radius')
conv = RegressionConvergenceRate(study,
                                 norm=CellNorm(),
                                 fiducials={ 'density': 1, 'pressure':1, 'velocity':1 })

conv.plot('density')
plt.savefig("convergence.pdf")
```

Standard Python library

Exactpack leverages Matplotlib, the predominant Python plotting package.

Load Exactpack components

Data readers for LANL codes

List of file names for data produced by codes

Reference solver for exact solution

Perform a convergence analysis using linear regression

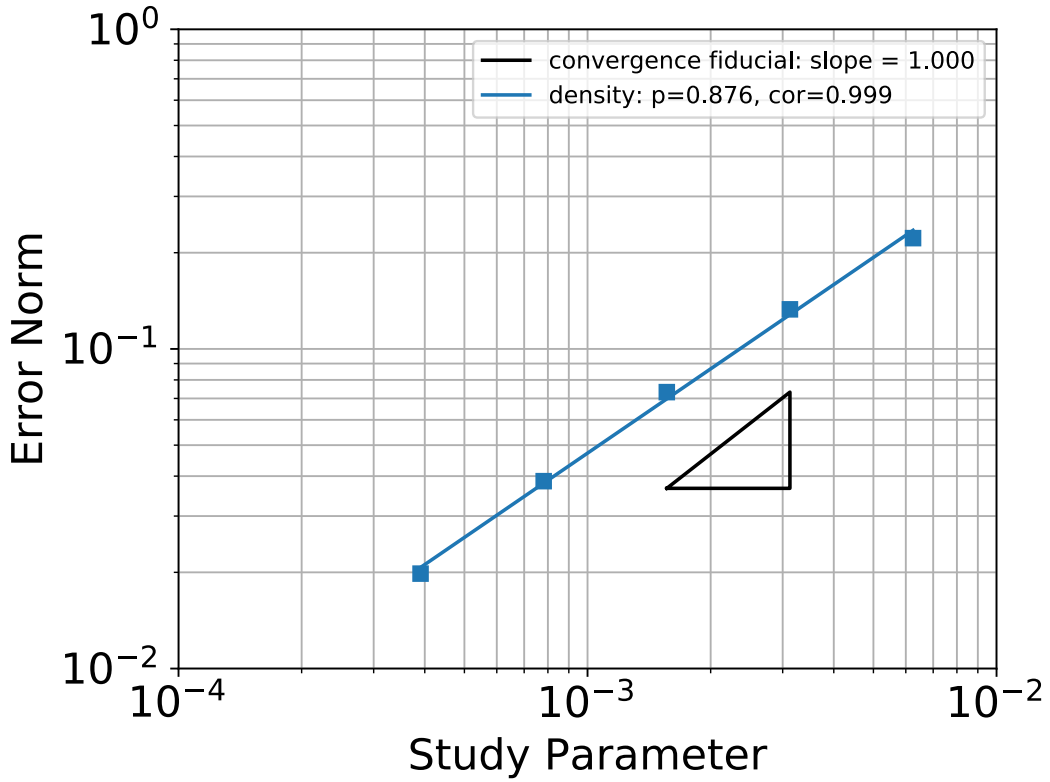
Volume weighted $L_1$ norm

Standardized plot of the results

Expected convergence rates

Save to file

# A sample convergence plot from ExactPack




***We will discuss the lessons from this result in the next talk.***

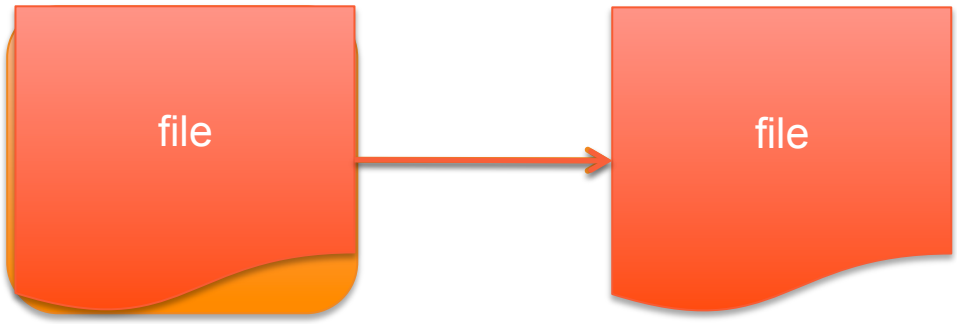
# Simulation Process and Data Management

- **SPDM is an emerging category of tools to manage pedigree and archive simulation data.**
  - Using SPDM all the inputs and important outputs from each simulation run are captured and archived.
  - Inter-process dependencies are tracked, so output pedigrees are available, and out-of-date results can be recomputed.
  - Studies can be performed easily by copying an existing result, making changes, and updating.
- **We considered some commercially available solutions.**
  - MSC Software, ANSYS, Seimens, EASA Software, etc.
  - Expensive, require significant computing resources, not well designed for our cyber-security profile
- **Our conclusion: current commercial offerings are not worth the trouble.**

# Lightweight Simulation Process and Data Management

- **LSPDM is designed to be serverless and easy to deploy.**
  - Installs as Python package
  - Does not require administrator privileges
- **Built on Subversion for version control**
  - Distributed version control cannot be used because there is too much data.
  - Narrow and shallow checkouts are a necessary requirement.
- **Simple command line interface is designed to be as close to what users do currently.**
  - Python API is scriptable, could be extended to include a GUI
- **Supports arbitrary scripts (Bash, Csh, Python, sed, etc.)**
- **Run interactively or LSPDM can control SLURM batch submission.**
- **Run all out-of-date processes in a dependency tree with one command.**

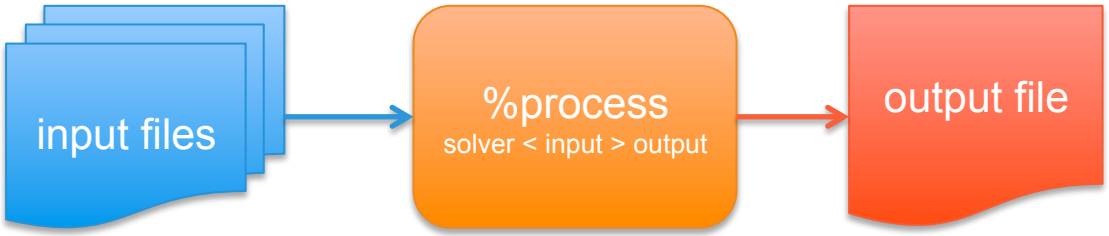
# The basic building blocks of LSPDM are files and processes




- **Any file type can be tracked as a file.**
  - Files can have no dependencies…
  - Copy dependencies…
  - Or depend on a process.

- **Files don't need to be explicitly added to the repository, they are automatically added by LSPDM when they are referred to.**

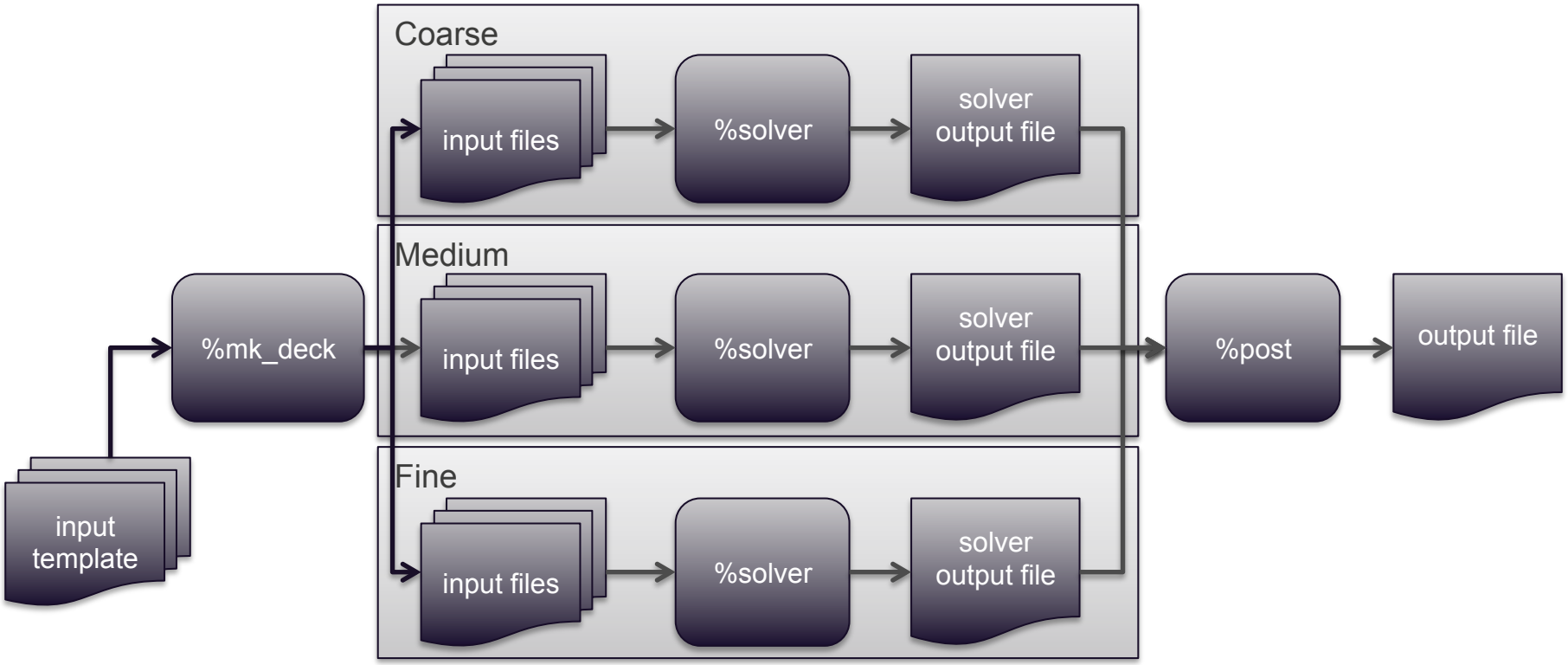
# The basic building blocks of LSPDM are files and processes




- **A process is a script that LSPDM can run.**
  - Processes have input dependencies...
  - Output dependencies are files which depend on that process.

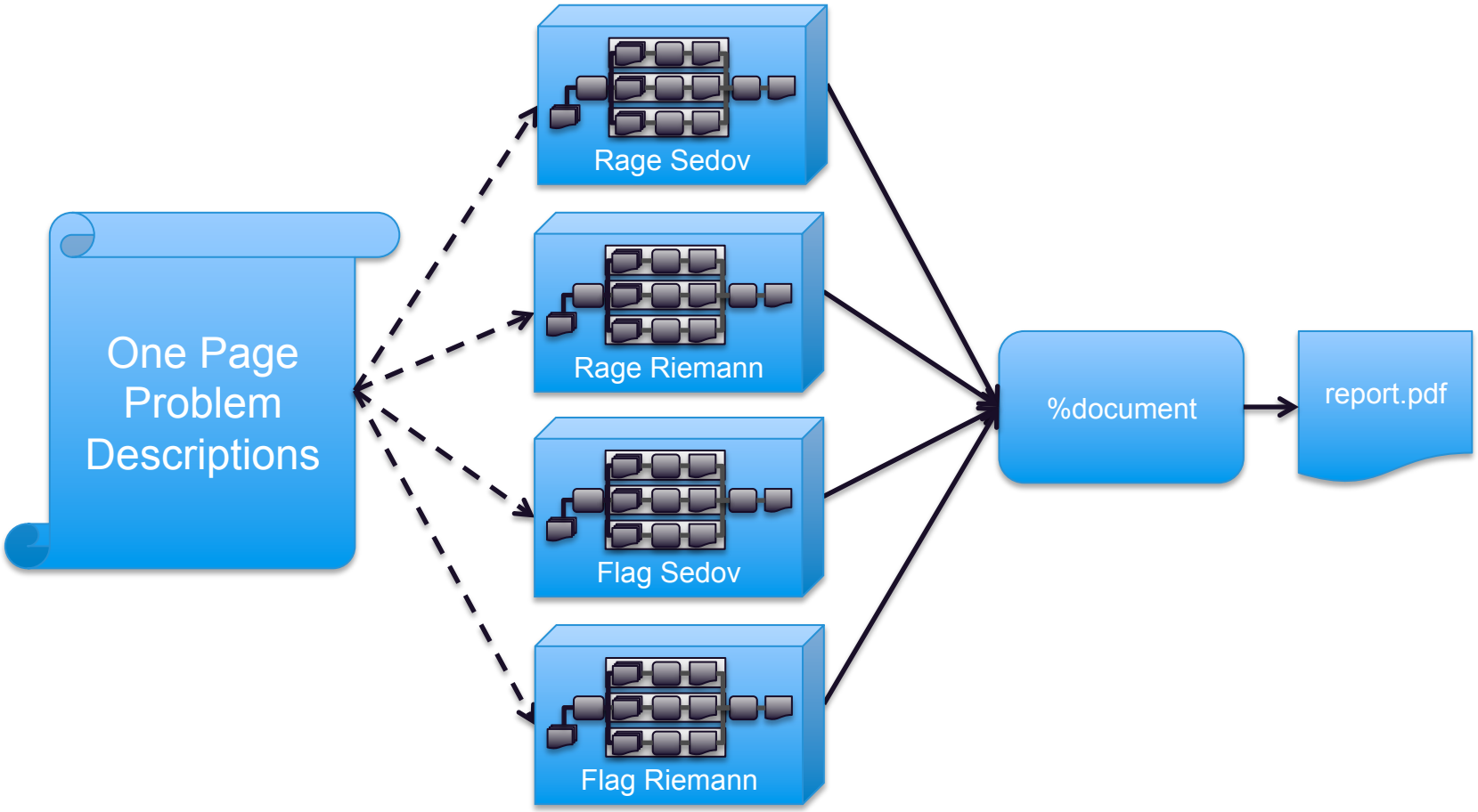
```
lspdm add %process \
        --cmd “solver < input > output” \
        --inputs “input files” \
        --outputs “output files”
```

# The basic building blocks of LSPDM are files and processes




- **These building blocks can be built up into complex dependency trees.**
- **If one file is changed, all dependencies can be updated.**

# LANL Verification Test Suite



# Conclusions

- We have put significant effort to develop infrastructure tools that support a pedigreed, repeatable, and shareable workflow.
- Whenever possible our tools are open source.
- We have received considerable interest in our tools from outside the Laboratory, as well as across the complex.
- For some of the results we have produced using our tools, stick around for the next talk.